Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1993	Professional Paper

4. TITLE AND SUBTITLE

EVOLVING RECURRENT PERCEPTRONS

5. FUNDING NUMBERS

PR: ZW67
PE: 0601152N
WU: DN303002

6. AUTHOR(S)

J. R. McDonnell and D. Waagen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Command, Control and Ocean Surveillance Center (NCCOSC)
RDT&E Division
San Diego, CA 92152-5001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This work investigates the application of evolutionary programming, a multi-agent stochastic search technique, to the generation of recurrent perceptons (nonlinear IIR filters) for time-series prediction tasks. The evolutionary programming paradigm is discussed and analogies are made to classical stochastic optimization methods. A hybrid optimization scheme is proposed based on multi-agent and single-agent random optimization techniques. This method is then used to determine both the model order and weight coefficients of linear, nonlinear, and parallel linear-nonlinear next-step predictors. The AIC is used as the cost function to score each candidate solution.

Published in *Proceedings of SPIE Science of Artificial Neural Networks II*, Vol. 1966.

14. SUBJECT TERMS

Neural Networks
Evolutionary Programming
Signal Detection

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE (include Area Code)	21c. OFFICE SYMBOL
J. McDonnell	(619) 553-5762	Code 731

Accesion For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

Evolving recurrent perceptrons

John R. McDonnell & Don Waagen

Naval Command, Control and Ocean Surveillance Center, RDT&E Div.
San Diego, CA 92152-5000

ABSTRACT

This work investigates the application of evolutionary programming, a multi-agent stochastic search technique, to the generation of recurrent perceptrons (nonlinear IIR filters) for time-series prediction tasks. The evolutionary programming paradigm is discussed and analogies are made to classical stochastic optimization methods. A hybrid optimization scheme is proposed based on multi-agent and single-agent random optimization techniques. This method is then used to determine both the model order and weight coefficients of linear, nonlinear, and parallel linear-nonlinear next-step predictors. The AIC is used as the cost function to score each candidate solution.

1. INTRODUCTION

Networks with recurrent connections represent an alternative to feedforward networks for predicting and classifying nonlinear time-series data. Recurrent networks can be transformed to purely feedforward networks by unfolding in time, where additional layers correspond to each time step. This technique may be prohibitive if the time structure is not well known or may become very complicated if the topology is dynamic. As a result, training can be greatly complicated by the addition of recurrent units in a dynamic artificial neural network. This work investigates the application of multi-agent stochastic search in determining the number of recurrencies as well as the weight coefficients of a single perceptron. A recurrent perceptron can be used as a fundamental node in recurrent network. The "perceptron" in this case simply refers to an infinite impulse response (IIR) filter with potentially nonlinear outputs. The application domain in this study is constrained to next-step prediction problems.

Simultaneously determining both perceptron weight coefficients and structure requires a search procedure that is amenable to combinatorial optimization problems. The more successful algorithms for these types of problems have generally been stochastic search techniques such as simulated annealing[1], genetic algorithms[2], and simulated evolution[3]. The simulated evolution, or evolutionary programming (EP), paradigm has been shown to have the desired attributes: combinatorial optimization capabilities[4], the ability to determine model structure[5], and the ability to train neural networks[6].

Crick and Asanuma[7] indicate that recurrency is the "general rule" as reciprocal connections are generated for projections from one cortical area to another cortical area. In partially recurrent artificial neural networks, the units which receive feedback are generally referred to as "context" units[8] since they provide information about past activation levels of the output or the hidden units. Williams[9] characterizes recurrency based on its utilization in a connectionist architecture. *Conservative recurrence* corresponds to a tapped-delay input signal. This approach yields a network which is sensitive to temporal patterns without directly incorporating recurrent units. This technique has been widely applied in the field of speech recognition (*c.f.*, Waibel *et al.*[10]). *Liberal recurrence* is the feedback from the output to the input units such as implemented by Jordan[11]. *Radical recurrence* encompasses both conservative and liberal recurrency as well as arbitrary recurrency among the hidden units. Elman's[12] architecture is representative of radical recurrency.

This work applies recurrency to a single processing unit similar to the adaptive linear combiners discussed by Widrow and Stearns[13]. A significant difference is that nonlinearities are imposed on the output of the linear combiners. If a hyperbolic tangent is used as the nonlinearity, then the linear combiner becomes a recurrent perceptron or an IIR filter with nonlinear output transformations. This investigation takes advantage of the EP framework to evolve the number of tapped-delay inputs as well as the number of tapped-delay feedbacks. Determining the model order of ARMA processes using EP was done by Fogel[5]. Recurrent structures have been successfully trained using EP by Angeline *et al.*[14], Saravanan[15], and McDonnell and Waagen[16]. Gradient methods for training recurrent nets are given by Rumelhart *et al.*[17] and Williams and Zipser[18].

The application domain investigated in this work is time-series prediction. Feedforward networks have been used with success for both system modeling and prediction . Narendra and Parthasarathy[19] used feedforward networks for system identification and control. Jones *et al.*[20] have formulated the Connectionist Normalized Linear Spline Network (CNLS) for time-series prediction. The CNLS is a feedforward network that incorporates Gaussian activations on the hidden units and normalizes the basis functions during training. Weigend *et al.*[21] use weight-elimination on feedforward networks to prevent overfitting the training data for a prediction task. Hinton *et al.*[22] have demonstrated the use of weight-sharing for training feedforward networks with prediction capabilities.

The next section describes Solis&Wets' random optimization technique and the general evolutionary programing algorithm. Variants of both methods are applied to finding extrema of an unknown function. A hybrid strategy is subsequently developed which embeds Solis&Wets' technique within the EP framework. Results of the hybrid approach for function optimization are also provided. The structure of the recurrent perceptron is then discussed. Finally, results are given for next-step prediction performance on some standard time-series benchmarks.

2. MULTI-AGENT STOCHASTIC SEARCH

2.1. Single-Agent Stochastic Search

Random optimization has traditionally been based on single-agent stochastic search (SASS) strategies. Karnop[23] discusses the benefits of using SASS strategies including the discovery of features of the cost surface. Both Karnop[23] and Rao[24] generate a random walk sequence to an extremum by perturbing the search point with a uniform random variable. Rao exploits the directionality of the randomly generated vectors which continue to yield lower valued objective functions. In a similar algorithmic formulation as Rao, Matyas[25] utilizes Gaussian perturbations about the search point. Solis & Wetts[26] have upgraded Matyas' random optimization approach by incorporating a bias in the mutation operator and evaluating the objective function at x-δx if evaluation at $x+\delta x$ does not improve the current value of the objective function. Baba[27] has successfully applied this SASS technique to training static networks. Simulated annealing[28] is also another example of SASS. The probabilistic nature of setting $x=x+\delta x$ if $f(x+\delta x) > f(x)$ provides a hill-climbing capability not contained in the methods previously discussed.

The basic algorithm formulated by Solis & Wets[26] is described as follows

1. *Initialize the search vector* x_0, *set* $k = 0$ *and* $b = 0$.
2. *Generate a Gaussian random vector* $\xi_k \sim N(b,\sigma)$.

3(a). *If* $f(x_k + \xi_k) < f(x_k)$, *then set* $x_{k+1} = x_k + \xi_k$ *and* $b_{k+1} = 0.4\xi_k + 0.2b_k$

3(b). *If* $f(x_k - \xi_k) < f(x_k) < f(x_k + \xi_k)$, *then set* $x_{k+1} = x_k - \xi_k$ *and* $b_{k+1} = b_k - 0.4\xi_k$

3(c). *Otherwise,* $x_{k+1} = x_k$ *and* $b_{k+1} = 0.5b_k$

4. *If* k = *maximum number of iterations then stop, else* $k=k+1$ *and go to Step 2.*

In the basic algorithm, the variance on ξ is controlled by the repeated number of successes or failures in decreasing the objective function f. The contraction and expansion constants, as well as the upper limits on the allowable number of trials, are set by the user.

This technique was modified so that the variance of the Gaussian perturbations are proportional to the magnitude of the objective function $\xi \sim N(b, f(x))$. This optimization process was applied to the Bohachevsky function $f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$. The transcendental terms cause many local minima inside the interval $x \in [-1,1]$ while the quadratic terms dominate the surface structure outside of this interval. A global minimum exists at $x=(0,0)$. The modified Solis & Wets optimization technique was used for finding the global minimum on the Bohachevsky surface as shown in Fig. 1 for the average of ten sessions. It is interesting to note that an average of 150 function evaluations were conducted for each session. As a comparison, the unmodified version of the basic algorithm was used to find the global minimum of the Bohachevsky function with the average results of ten sessions shown in Fig. 2. On average, each optimization session contained 189 function evaluations. It should be noted that in one instance, the algorithm became trapped at a local minima. The variance parameters are the same as those given by Solis & Wets[26].

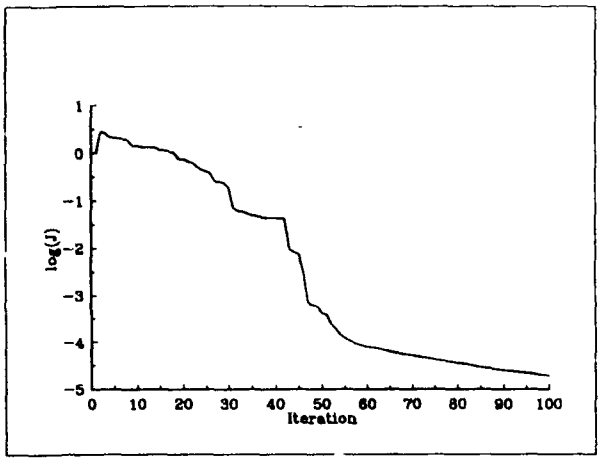

Fig. 1. The modified Solis & Wets technique applied to finding the global minimum of the Bohachevsky funcntion. Averaged over ten trials.

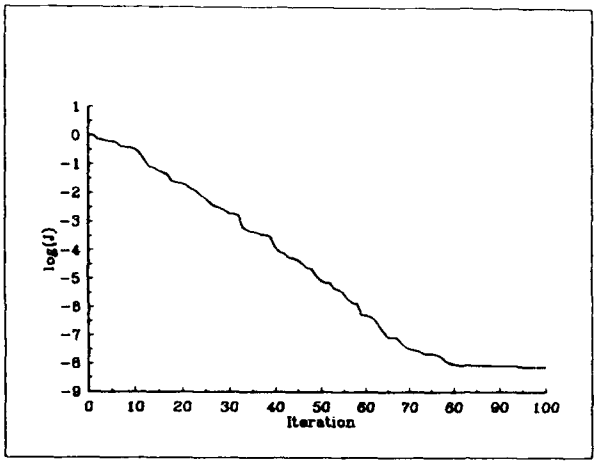

Fig. 2. The unmodified Solis & Wets technique applied to finding the global minimum of the Bohachevsky function. Averaged over ten trials.

The power of the unmodified Solis & Wets technique appears self-evident as illustrated by Figures 1 and 2 where *log(J)* is the $log_{10}(f(x))$. The observation might be made that reducing the variance independent of the objective function is beneficial to the search. It should be noted that the lower bound on the standard deviation of the random perturbation for the unmodified method was set at $\sigma_{lb} = 10^{-4}$. Higher precision might have been attained (on average) at the expense of more frequent entrapment in local minima. To avoid entrapment conditions it is suggested that a global search agent be employed in parallel with the local method.

2.2. The Evolutionary Programming Paradigm

In 1958, Brooks[29] described a *creeping random method* where *k* points were generated via Gaussian perturbations about a search point. The best point was kept and the process repeated. Brooks observed that "there are some rather intriguing analogies that can be made between the creeping random method and evolution". This analogy was also apparent to Fogel *et al.*[3] who proposed a multi-agent search strategy incorporating a population of organisms that are mutated to yield offspring. The resulting search strategy was termed *evolutionary programming* (EP).

EP is a multi-agent stochastic search (MASS) paradigm used for finding global extrema. Although the EP methodology simulates the evolutionary process found in nature, the mechanisms incorporated in this framework and resulting characteristics may also be found in some of the stochastic optimization techniques previously discussed. The perturbation applied in the search is typically a multivariate normal random variable $\delta x \sim N(0, S_f \cdot J)$ where S_f is the scale factor and J is the magnitude of the objective function. Hill-climbing and tunneling are achieved though the application of the mutation operator in concert with the multi-agent capabilities of EP. Analogous to the hill-climbing ability of simulated annealing relaxation methods, EP employs a competition process which allows less fit organisms (search points) to be retained in the population in a probabilistic fashion. The EP optimization algorithm can be described by the following steps[5]

1. Form an initial population $P_{2N-1}(x)$ of size 2N. The parameters x associated with parent element P_i are randomly initialized from a user specified search domain.
2. Assign a cost to each element $P_i(x)$ in the population based on the associated objective function J_i.
3. Reorder the population based on the number of wins generated from a stochastic competition process.
4. Generate offspring (P_N .... P_{2N-1}) from the highest ranked N elements (P_0 .... P_{N-1}) in the population by perturbing x with a multivariate normal random variable $\delta x \sim N(0, S_f \cdot J)$.
5. Loop to step 2.

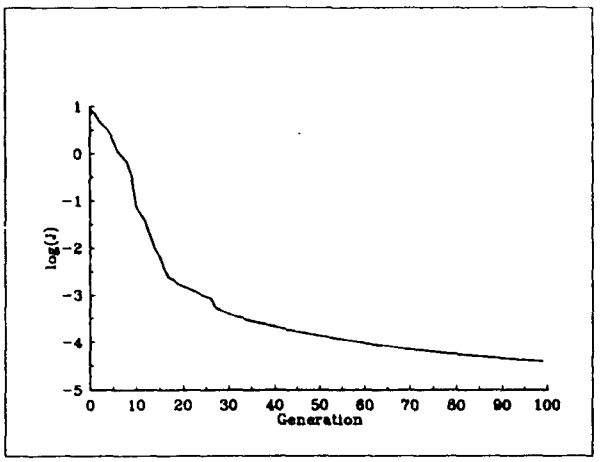

Fig. 3. Optimization of the Bohachevsky function using the EP algorithm, S_f=1, 50 parents, 20 competitions.

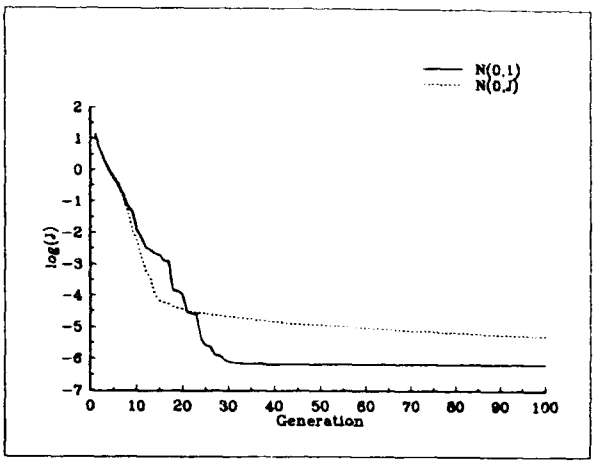

Fig. 4. Augmenting EP with bisection search. Half of offspring were generated by mutation, the other half by recombination.

The EP algorithm was applied to the Bohachevsky function with the results shown in Fig. 3. For these experiments, the scale factor was set as $S_f=1$ for 50 parents with one offspring each and ten runs were made. There were 20 competitions held for each member of the population. A bisection search was implemented by allowing randomly chosen parents to recombine according to $x_o = 0.5(x_i + x_j)$ where x_i and x_j are the randomly chosen parent vectors and x_o is the offspring vector. This was done for half the offspring while the other half were generated using the perturbation approach $\delta x \sim N(0, J)$. Nearly an order of magnitude improvement was observed as shown in Fig. 4. The next experiment decoupled the cost function from the perturbation size so that $\delta x \sim N(0,1)$. An order of magnitude improvement was observed as shown in Fig. 4. Fogel[5] reports that it took an average of 65.5 generations to achieve $log_{10}(f(x)) < 10^{-6}$ using the same number of parents and offspring in the general EP method. By decoupling the cost function and implementing a simple bisection search, it took less than 30 generations, on average, to achieve similar results.

2.3. A Hybrid Approach

A variant of the EP search strategy is proposed to take advantage of the efficiency of convex optimization (such as the bisection search) as well as the global search capability provided by MASS strategies. From the previous experiments the following capabilities appear to be beneficial to a hybrid approach: multi-agent search tends to avoid local minima; if the offspring does not yield a lower cost, then check in the other direction; convex optimizaton and perturbation variance reduction improves precision. Making the perturbation variance proportional to the value of the cost function does not appear to provide significantly better results and may even inhibit the rate of convergence with respect to other approaches. Decoupling the perturbation variance from the cost function value may prove beneficial since oftentimes the shape of the search surface is not well known and may even take on negative values. A similar strategy was employed by Waagen *et al.*[30] in the formulation of the *stochastic direction set* method.

Fig. 5 illustrates a hybrid approach which generates a variety of different offspring within the EP framework. The first set of offspring are generated to explore the search space in a global fashion. These offspring mutate the parent search points with the perturbation $\delta x \sim N(0,\sigma)$ where σ is fixed. The second set of offspring implement convex optimization through recombination. This may be as simple as the bisection method used above. The final set of offspring are generated using the method of Solis & Wets. This set of offspring replace the parent organisms since they are equivalent to or better than their precursors. The bias term which provides a momentum to the search must also be included. Offspring generated by the first method have the bias term set to zero. The recombination rules which apply to the second set of offspring can also be applied to other parameters including the bias term β.

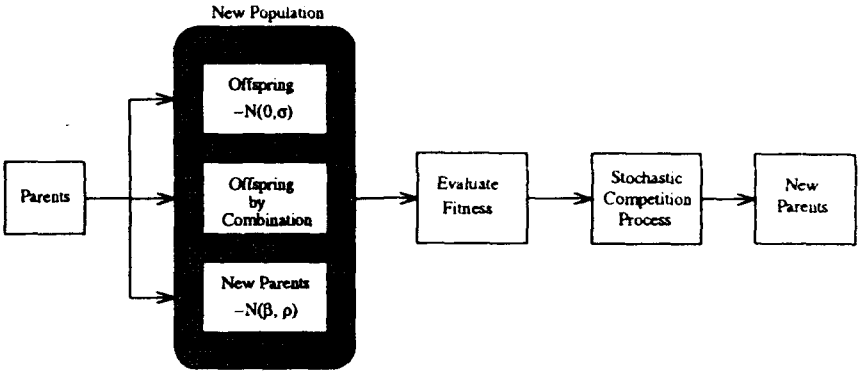

Fig. 5. One generation of the hybrid multi-agent stochastic search.

A variety of other techniques may be employed as alternatives to the stochastic competition process. Saravanan[15] replaces the least fit organism in the population with each additional offspring. Another strategy might be to retain less fit organisms by disallowing competition across different levels of maturity. The maturity of an organism could be determined by how many generations it has existed within the population. A deterministic means to minimize redundancy might also be employed. To delay the potential dominance of a single organism in the population, the number of competitions can be set to an arbitrarily low value. This allows the retention of less fit organisms, thus providing a more exhaustive search. As the number of competitions increases, the retention of the best fit individuals becomes more deterministic.

The hybrid approach was employed on the Bohachevsky function with the average cost from ten trials shown in Fig. 6. For this particular example, it appears that the hybrid technique improves the efficiency of the search from both precision and convergence aspects. As shown in Fig. 6, the hybrid technique attained 10 orders of magnitude precision within 50 generations (which corresponds to a maximum of 7550 function evaluations). A comparison between Solis&Wets' technique and the hybrid approach which utilizes their method was also made for the Rosenbrock function. This function is referred to as a banana valley since it contains a steep valley along $x_2 = x_1^2$. The Rosenbrock function is given by $f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1^2)^2$. Fig. 7 shows the average and best results from ten trials for both the Solis & Wets technique and the hybrid approach outlined above. To compare the search processes based on the number of function evaluations, each generation is equivalent to a maximum of 150 function evaluations for the Solis & Wets method and a maximum of 150 function evaluations for the hybrid approach. Both average curves show optimization is still occurring after the maximum number of generations or iterations were reached and the experiment stopped.

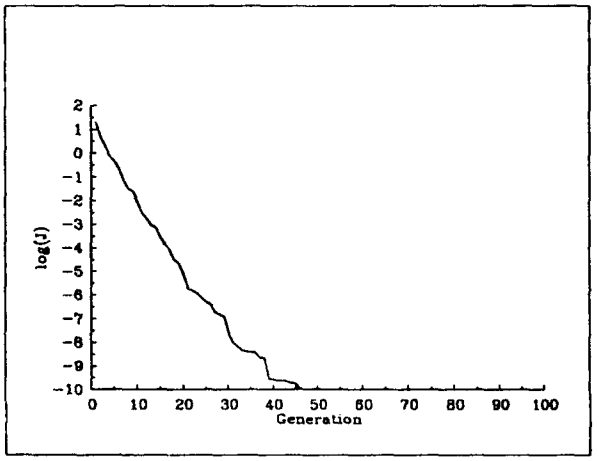

Fig. 6. Optimization of the Bohachevsky function using the hybrid strategy and 50 search agents.

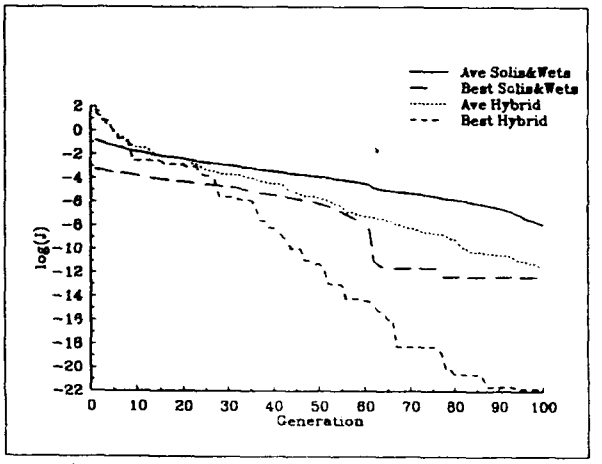

Fig. 7. Comparing the Solis & Wets and hybrid approach on the Rosenbrock function.

3. EVOLVING PERCEPTRONS

This section investigates the application of EP to evolving a recursive adaptive linear combiner[13] with a nonlinear output or activation function. The neuron model will serve as a next-step predictor and is given by

$$\hat{y}(k+1) = f\left(\sum_{i=0}^{m-1} a_i y(k-i) + \sum_{j=1}^{n-1} b_j \hat{y}(k-j+1)\right)$$

where the search strategy must determine the order of the feedforward terms, m, the order of the feedback terms, n, as well as the feedforward coefficients, a_i, and the feedback coefficients, b_j. This structure is shown in Fig. 8.

The sigmoid function is a candidate for the nonlinear mapping since it approximates the perceptron's output as an inverted polynomial series

$$f(x) = (1+e^{-x})^{-1} \approx \left(1+\sum_{i=0}^{n} \frac{(-x)^i}{i!}\right)^{-1}$$

Other nonlinear functions may be equally applicable as discussed by Cybenko[31]. For example, a polynomial series is generated by

$$f(x) = \cos(x) + \sin(x) \approx 1 + x - \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} \ldots$$

or, the search may even be conducted over a set of candidate mapping functions F such that $f \in F$.

The cost function for each perceptron is given by the Akaike information criterion (AIC)[32]

$$AIC(m,n) = N \ln(\hat{\sigma}_e^2) + 2(m+n+1)$$

where N is the effective number of observations. An additional factor of 2 is added to the number of fitted parameters $(m+n-1)$ since m and n must also be determined by the search strategy. If a bias term b_0 is added to the perceptron, then the number of fitted parameters must be increased to $(m+n+2)$. The MLE of the innovation variance is determined according to

$$\hat{\sigma}_e^2 = \frac{1}{N}\sum_{k=0}^{N-1} \hat{e}^2(k)$$

If it is desired that direct linear feedthrough (DLF)[33] capabilities be present in parallel with the nonlinear contributions, then the perceptron structure can be reformulated as a combination of linear and nonlinear recurrencies. This idea is express by

$$\hat{y}(k+1) = \hat{y}_L(k+1) + \hat{y}_N(k+1)$$

where

$$\hat{y}_L(k+1) = \sum_{i=0}^{m-1} a_i y(k-i) + \sum_{j=1}^{n-1} b_j \hat{y}(k-j+1)$$

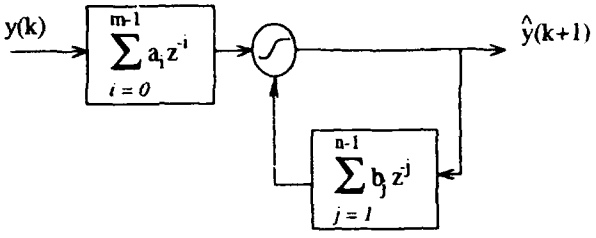

Fig. 8. The recurrent perceptron structure used for next-step prediction.

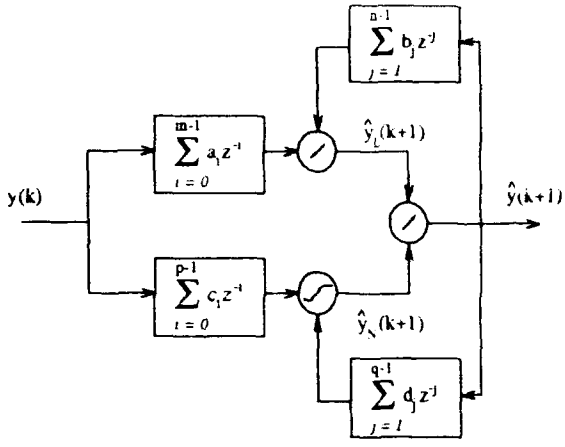

Fig. 9. The parallel linear-nonlinear recurrent structure used for next-step prediction.

$$\hat{y}_N(k+1) = f\left(\sum_{i=0}^{p-1} c_i y(k-i) + \sum_{j=1}^{q-1} d_j \hat{y}(k-j+1)\right)$$

If this structure is used, then the AIC score is described by

$$AIC(m,n,p,q) = N \ln(\hat{\sigma}_e^2) + 2(m+n+p+q+2)$$

The number of fitted parameters is *(m+n+p+q+4)* if both b_0 and d_0 biases are included. The parallel linear-nonlinear structure is shown above in Fig. 9.

4. PREDICTION RESULTS

4.1. Predicting Sunspots

The first set of experiments were conducted on Wolf's sunspot series over the years 1700-1983. These numbers are indicative of the average relative number sunspots observed each day of the year. Consistent with Weigend *et al.*[21], sunspot data from 1700-1920 was used for training and data from 1921-1983 was used for testing. All of the data was normalized by a factor of 200. A small number of experiments were run for 1000 generations with 20 parents, 20 offspring (10 for convex optimization and 10 for global mutation), 10 competitions, and $\rho_{lb} = 0.01$. The maximum possible model order was equivalent for both the feedforward and feedback lines $m_{max}=n_{max}=p_{max}=q_{max}=21$. The results are given in Table I. Even though the AIC criterion is used, some degree of overfitting may still occur as indicated by the second entry in Table I. This experiment yielded the lowest MSE on the training set but did not generalize as well as other results on the test set.

The solutions which yielded the best combined test+training AIC scores are shown in Figs.10-13. Figs. 10 and 11 show the training and test results, respectively, for a parallel linear-hyperbolic tangent 8-3-10-3 implementation. A linear bias was also incorporated for this trial. Figs. 12 and 13 show the training and test results, respectively, for a simple linear implementation of Fig. 8. Again, a bias term was incorporated in this experiment. It is interesting to note that the evolved structure for the linear system did not include recurrency and relies only on the three previous observations and a bias term. Another interesting observation is that there is only a slight difference between the resulting prediction curves shown in Figs. 11 and 13. The better training+test AIC scores are comparable to the AIC scores given by Priestley[34] for AR and ARMA process models. As another comparison, a pure follower strategy where the next-step prediction is just the current value yields an AIC= -1192.5 on the first 278 data points.

functions	m-n-p-q	MSE: Training	MSE: Training+Test	AIC: Training	AIC: Training+Test
linear+bias+sin	6-5-11-4	0.00508	0.00588	-982	-1278
linear+bias+sin	4-4-2-2	0.00470	0.00678	-1025	-1268
linear+bias+tanh	8-3-10-3	0.00498	0.00566	-990	-1291
linear+bias+ tanh(w/ bias)	7-3-7-5	0.00489	0.00597	-996	-1280
linear+bias	3-0-x x	0.00574	0.00686	-1006	-1285

Table I. Results from sample experiments evolving next-step sunspot predictors.

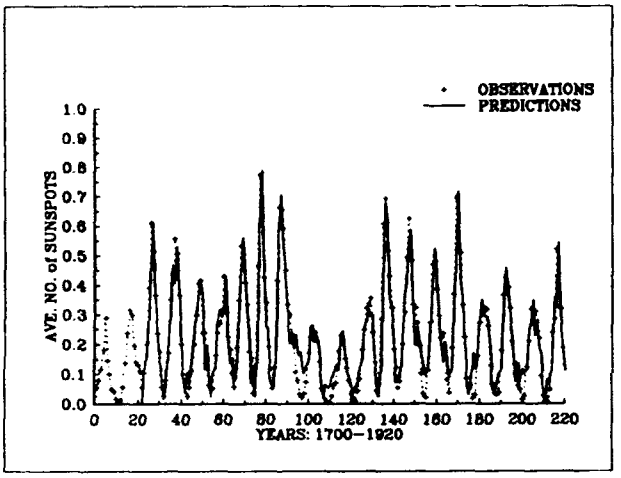

Fig. 10. The sunspot training results for f=linear+bias+ tanh with structure 8-3-10-3.

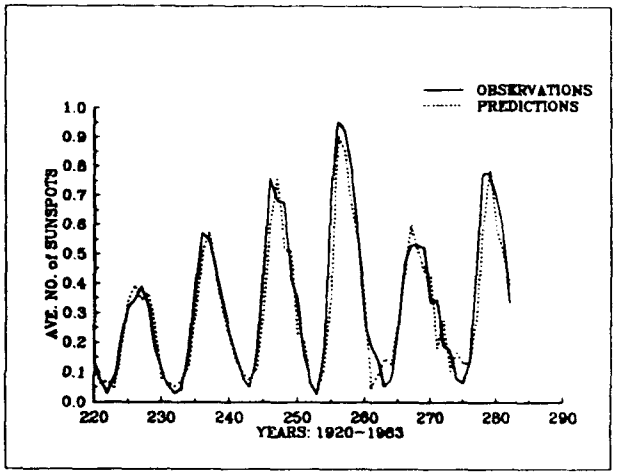

Fig. 11. The sunspot test results for f=linear+bias+ tanh with structure 8-3-10-3.

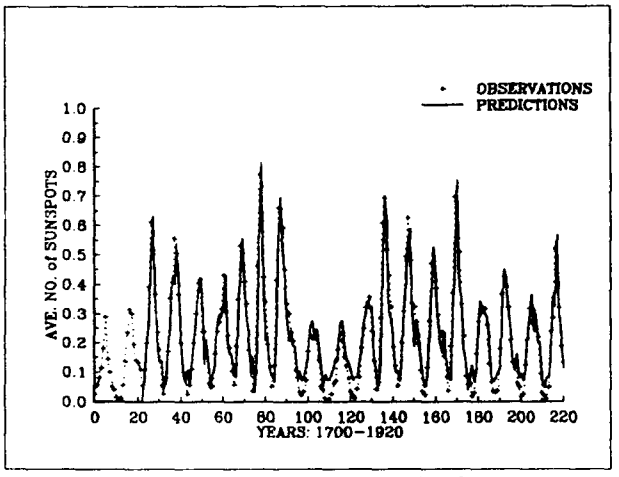

Fig. 12. The sunspot training results for f=linear+bias with structure 3-0.

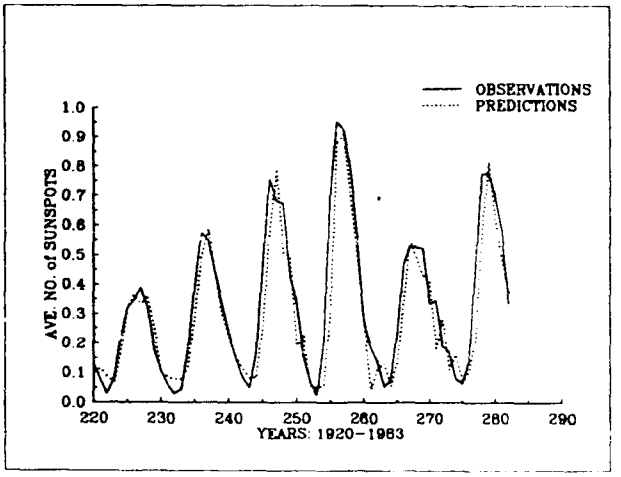

Fig. 13. The sunspot test results for f=linear+bias with structure 3-0.

4.2. The Iterated Quadratic Map

Weigend *et al.*[21] use the iterated quadratic map $x_{k+1} = 4x_k(1-x_k)$ on the unit interval as an example of deterministic chaos. This case in point precludes any long-term predictability although short term predictions are possible. Weigend *et al.*[21] indicate that, on average, predictive performance is lost after n iterations where n is the number of bits representing x_k. This problem is essentially an interpolation in state space as demonstrated by Fig. 14. The same parameters were used as in the previous experiment. Fig. 15 shows the performance of the evolved predictor $\hat{x}_{k+1} = \sin(3.1476x_k) + 0.0274$ on the training set after 5000 training generations. This model yields an AIC score of -1033 and an MSE=0.00533 on the training+test data. This solution was evolved from a parallel arrangement where one node implements the *cos* mapping and the parallel node implements the *sin* mapping. The state-space plot in Fig. 14 designates the evolved solution as marked by the '*'s. The test results are shown in Fig. 16. Fig. 17 shows the test results if the predictor takes the form $\hat{x}_{k+1} = \sin(\pi x_k)$. The state-space plot of this solution is designated by the 'o's in Fig. 14. As in the previous example, a solution was found which did not rely on recurrency to approximate the time-series.

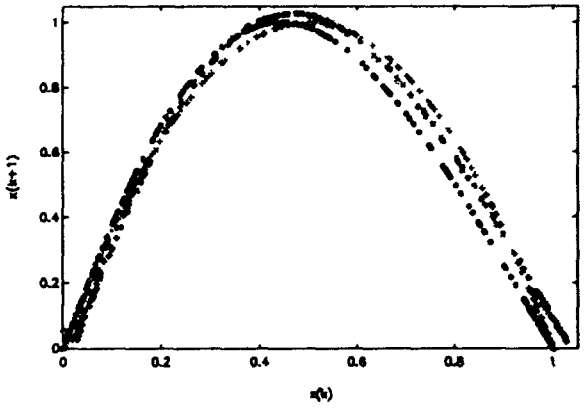

Fig. 14. State-space plot of observed data (+), evolved solution (*), and an estimate (o) generated by inspection of the evolved solution.

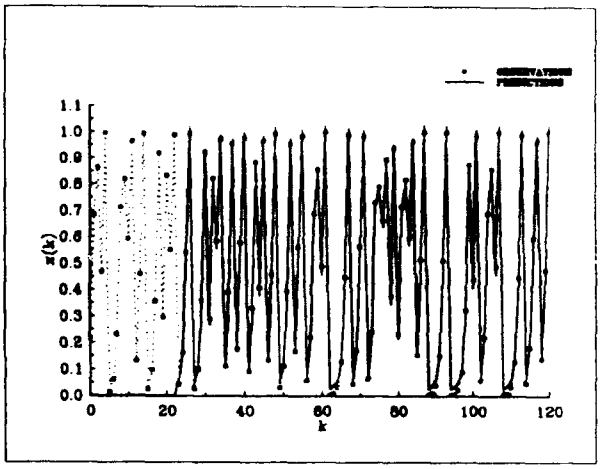

Fig. 15. Training data and the results generated from the evolved solution $\hat{x}_{k+1} = \sin(3.1476x_k) + 0.0274$.

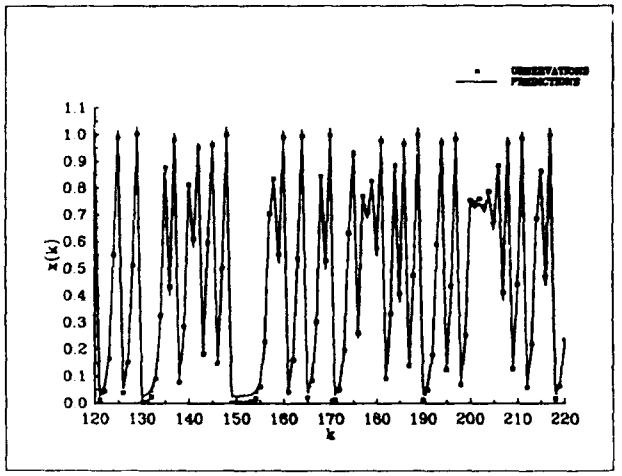

Fig. 16. Test data and the evolved solution of the form $\hat{x}_{k+1} = \sin(3.1476x_k) + 0.0274$.

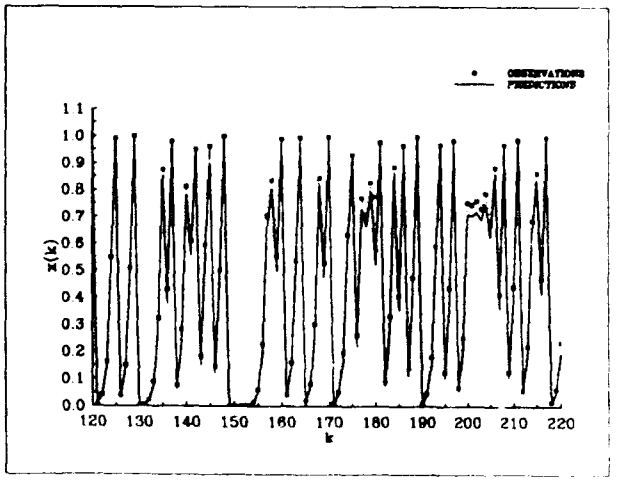

Fig. 17. Test data and the estimate $\hat{x}_{k+1} = \sin(\pi x_k)$ generated from inspection of the evolved solution.

4.3. The Mackey-Glass Equation

The Mackey-Glass equation represents a model for white blood cell production in leukemia patients[35]. This model is complicated by the addition of a time delay τ in the nonlinear differential equation

$$\dot{x}(t) = \frac{ax(t-\tau)}{1+x^c(t-\tau)} - bx(t)$$

where the free parameters are set as $a=0.2$, $b=0.1$, $c=10$, and $\tau=30$ according to Jones *et al.*[36]. In all experiments 500 data points were in the training set with the test set comprising the subsequent 500 data points. All of the data was normalized by a factor of 1.4. The experiments also used the parallel sin-cos nodal arrangement with a bias as used in the previous example. Fig. 18 shows the training data and the results from a 5-7-3-3 configuration. The training set trained to a MSE=0.0005 and AIC=-3609 after 5000 generations. The test+training set have a MSE=0.00028 and AIC=-7968. Fig. 19 shows the prediction capabilities of this model on the test set. A more challenging section of the Mackey-Glass equation was evaluated as shown in Fig. 20. A new model was generated and, for 1000 generations of training, a MSE=0.00109 and AIC=-3181 were attained on the first 500 points with a 10-7-8-8 configuration structure. For this model, a MSE=0.00066 and AIC=-7089 were achieved for the full sequence of test+training data. The error for the test and training sequences is shown in Fig. 21.

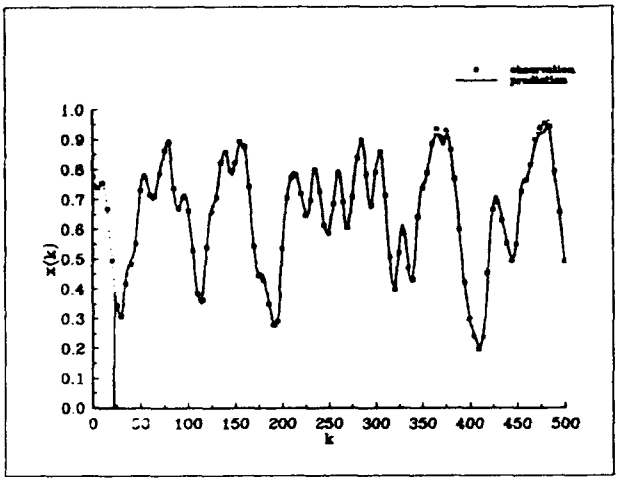

Fig. 18. The training results for a 5-7-3-3 configuration.

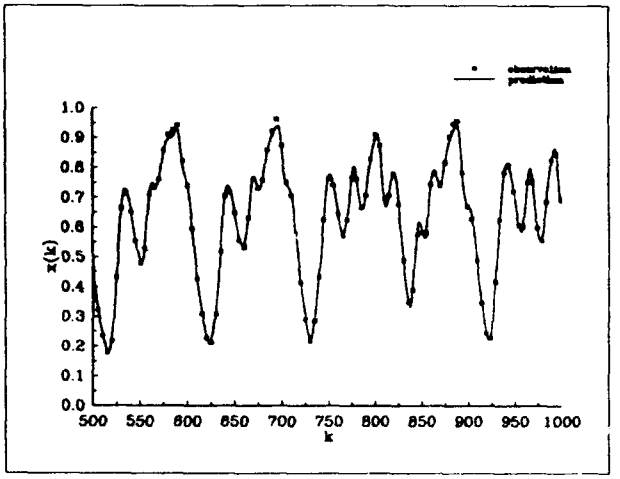

Fig. 19. The test results for the 5-7-3-3 configuration.

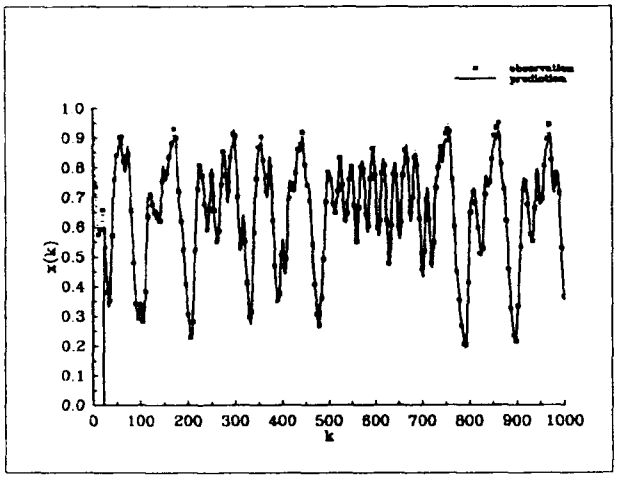

Fig. 20. The training and test results for a 10-7-8-8 configuration.

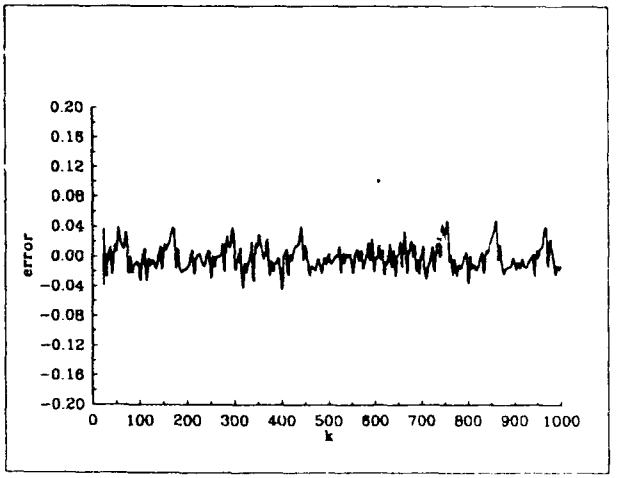

Fig. 21. The error for the sequence shown in Fig. 20.

5. CONCLUSION

This work has incorporated an efficient single-agent search strategy, the method of Solis&Wets, into the EP framework and augmented this with convex optimization capabilities to yield a hybrid multi-agent stochastic search technique. The method was applied to parallel linear-nonlinear IIR filters for next-step prediction tasks. The evolved solutions did not always have a recurrent structure and, as a result, simple implementations were found using this approach. Oftentimes, simplicity is not an option when a large non-dynamic architecture is specified for a given task. The lack-of-complexity of the evolved solutions for the standard problem set investigated in this work shows great promise for future implementations. It remains to be investigated as to how well this approach scales up to more complex data sets. The incorporation of the output transfer function into the search would be the next evolution of this work.

The simple recurrent structures investigated in this work demonstrated a high degree of capability for the time-series problems studied. Similar levels of proficiency were attained for a varied assortment of models. This might lead one to conclude that the joint parameter-function space is dense in the number of possible solutions, some of which are found by the relaxation scheme employed in this investigation. By cascading and/or parallelizing as in traditional feedforward neural network architectures, the recurrent nodes may present additional capabilities for more complex time-series processing tasks.

6. ACKNOWLEDGMENT

The authors thank Dr. Al Gordon, program director for NRaD internal research projects, for his support of this work.

7. REFERENCES

1. E. Aarts and J. Korst, *Simulated Annealing and Boltzman Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley & Sons, 1989.

2. J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, 1992.

3. L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons, 1966.

4. D. B. Fogel, "An evolutionary approach to the traveling salesman problem," *Biological Cybernetics*, Vol. 60, No. 2, 1988.

5. D. B. Fogel, *System Identification though Simulated Evolution: A Machine Learning Approach to Modeling*, Ginn Press, Needham, MA, 1991.

6. D.B. Fogel, L. J. Fogel, and V. W. Porto, "Evolving neural networks," *Biological Cybernetics*, Vol. 63, pp.487-493, 1990.

7. F. Crick and C. Asanuma, "Certain aspects of the anatomy and physiology of the cerebral cortex", in *Parallel Distributed Processing, Volume 1*, D.E. Rumelhart and J.L. McClelland (Eds.), MIT Press, 1986.

8. J. Hertz, A. Krogh, and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, 1991.

9. R.J. Williams, *Adaptive State Representation and Estimation using Recurrent Connectionist Networks*, MIT Press, Cambridge, MA, 1990.

10. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. 37, No. 8, 1989.

11. M.I. Jordan, "Serial order: a parallel distributed processing approach," Technical Report 8604, Institute for Cognitive Science, University of California, San Diego, 1986.

12. J.L. Elman, "Finding structure in time," Technical Report CRL 8801, University of California, San Diego, 1988.

13. B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice-Hall, 1985.

14. P. Angeline, G.M. Saunders, and J.B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," unpublished.

15. N. Saravanan, "Evolving neural networks: application to a prediction problem," *Second Annual Conf. on Evolutionary Programming*, San Diego, 1993.

16. J.R. McDonnell and D. Waagen, "Neural network structure design by evolutionary programming," *Second Annual Conf. on Evolutionary Programming*, San Diego, 1993.

17. D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation", in *Parallel Distributed Processing, Volume 1*, D.E. Rumelhart and J.L. McClelland (Eds.), MIT Press, 1986.

18. R.J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, Vol. 1, 1989

19. K.S. Narendra and K. Parthasarathy, "Identification and control of dynamic systems using neural networks," IEEE Trans. on Neural Networks, Vol. 1, No. 1, 1990.

20. R.D. Jones, Y.C. Lee, S. Qian, C.W. Barnes, K.R. Bisset, G.M. Bruce, G.W. Flake, K. Lee, L.A. Lee, W.C. Mead, M.K. O'Rourke, I.J. Poli, and L.E. Thode, "Nonlinear adaptive networks: a little theory, a few applications," Technical Report LA-UR 91-273, Los Alamos National Laboratory, Los Alamos, New Mexico,

21. A. S. Weigend, D. E. Rumelhart, and B.A. Huberman, "Predicting the future: a connectionist approach", Technical Report Stanford-PDP-90-01 or PARC-SSL-90-20, 1990.

22. S.J. Nowlan and G.E. Hinton, "Simplifying neural networks by soft weight sharing", *Neural Computation*, Vol. 4, No. 4, 1992.

23. D.C. Karnop, "Random search techniques for optimization problems," *Automatica*, 1, pp. 111-121, 1963.

24. S.S. Rao, *Optimization Theory and Applications*, John Wiley & Sons, 1979.

25. J. Matyas, "Random optimization," *Automation and Remote Control*, 26, 1965.

26. F.J. Solis and J.B. Wetts, "Minimization by random search techniques," *Mathematics of Operations Research*, 6, 1981.

27. N. Baba, "A new approach for finding the global minimum of error function of neural networks," *Neural Networks*, Vol. 2, 1989.

28. S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by simulated annealing," *Science*, 220, 1983.

29. S. H. Brooks, "A discussion of random methods for seeking maxima," *Operations Research*, 6, pp. 244-251, 1958.

30. D. Waagen, P. Diercks, and J.R. McDonnell, "The stochastic direction set algorithm: a hybrid technique for finding function extrema," *First Annual Conf. on Evolutionary Programming*, La Jolla, CA, 1992.

31. G. Cybenko, "Approximation by superpositions of sigmoidal functions," *Mathematics of Control, Signals*, and Systems. 1989.

32. H. Akaike, "A new look at the statistical model identification", *IEEE Trans. on Automatic Control*, Vol. 19, No. 6, Dec., 1974.

33. D. Haesloop and B.R. Holt, "Neural networks for process identification,", *Int. Joint Conf. on Neural Networks*, San Diego. 1992.

34. M.B. Priestley, *Non-linear and Non-stationary Time Series Analysis*, Academic Press, 1988.

35. M.C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, 197, 1977.

36. W.C. Mead, R.D. Jones, Y.C. Lee, C.W. Barnes, G.W. Glake, L.A. Lee, and M.K. O'Rourke, "Prediction of chaotic time series using CNLS-NET -- example: the Mackey-Glass equation," Technical Report LA-UR-91-720, Los Alamos National Laboratory. Los Alamos. NM. 1991.